**Unifesp/ICT**

**Arquitetura e Organização de Computadores**

**Profa. Denise Stringhini**

**Exercícios - Simulador 8 bits**

**(https://schweigi.github.io/assembler-simulator/)**

1. Baseado no exemplo inicial do simulador, imprima os caracteres em posição par da string.

; Simple example

; Writes Hello World to the output

JMP start

hello: DB "Hello World!" ; Variable

DB 0 ; String terminator

start:

MOV C, hello ; Point to var

MOV D, 232 ; Point to output

CALL print

HLT ; Stop execution

print: ; print(C:\*from, D:\*to)

PUSH A

PUSH B

MOV B, 0

.loop:

MOV A, [C+1] ; Get char from var

MOV [D], A ; Write to output

INC C

INC C

INC D

CMP B, [C] ; Check if end

JNZ .loop ; jump if not

POP B

POP A

RET ;Return

1. Baseado no exemplo inicial do simulador, imprima apenas os caracteres na posição ímpar da string.

; Simple example

; Writes Hello World to the output

JMP start

hello: DB "Hello World!" ; Variable

DB 0 ; String terminator

start:

MOV C, hello ; Point to var

MOV D, 232 ; Point to output

CALL print

HLT ; Stop execution

print: ; print(C:\*from, D:\*to)

PUSH A

PUSH B

MOV B, 0

.loop:

MOV A, [C] ; Get char from var

MOV [D], A ; Write to output

INC C

INC C

INC D

CMP B, [C] ; Check if end

JNZ .loop ; jump if not

POP B

POP A

RET ;Return

1. Baseado no exemplo inicial do simulador, imprima apenas os caracteres minúsculos.

; Simple example

; Writes Hello World to the output

JMP start

hello: DB "Hello World!" ; Variable

DB 0 ; String terminator

start:

MOV C, hello ; Point to var

MOV D, 232 ; Point to output

CALL print

HLT ; Stop execution

print: ; print(C:\*from, D:\*to)

PUSH A

PUSH B

MOV B, 0

.loop:

MOV A, [C] ; Get char from var

CMP A, 97 ; Compare with 'a' (ASCII value)

JB .not\_lowercase ; Jump if it's not a lowercase letter

CMP A, 122 ; Compare with 'z' (ASCII value)

JA .not\_lowercase ; Jump if it's not a lowercase letter

MOV [D], A ; Write to output

INC D

.not\_lowercase:

INC C

CMP B, [C] ; Check if end

JNZ .loop ; Jump if not

POP B

POP A

RET

1. Baseado no exemplo inicial do simulador, imprima a string de trás para a frente

; Simple example

; Writes Hello World to the output

JMP start

hello: DB "Hello World!" ; Variable

DB 0 ; String terminator

start:

MOV C, hello ; Point to var

MOV D, 232 ; Point to output

CALL print

HLT ; Stop execution

print: ; print(C:\*from, D:\*to)

PUSH A

PUSH B

MOV B, 0

.end:

MOV A, [C] ; Get char from var

INC C

CMP B, [C] ; Check if end

JNZ .end ; jump if not

.loop:

MOV B, 72

DEC C

MOV A, [C] ; Get char from var

MOV [D], A ; Write to output

INC D

CMP B, [C] ; Check if end

JNZ .loop ; jump if not

POP B

POP A

RET